1 Assignment 12 (100 points)

To submit your work, collect your inputs not provided as part of the assignment, the required outputs, and your answers to the written questions in pdf format (preferred) or Word, or .txt . Clearly label all of your work. For the math-oriented portion of the assignment, you may typeset in LaTeX or Word or provide a clear photograph of a hand-written solution. Work that is not legible will not be graded. Name a zip file `CS585_Assignment12_username.zip` and submit with web-submit. Please facilitate grading by labeling the code that you change with `//modified by deht`

1.1 Learning Objectives

- Understand the different entities involved in camera geometry, such as camera positions and rotations, world points, image points, epipoles, and epipolar lines
- Understand how these entities are related
- Understand which entities can be used to infer the other entities

1.2 Programming Assignment

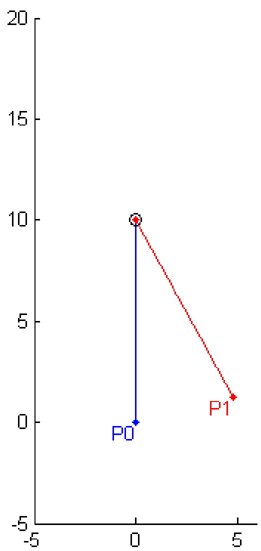
This assignment uses OpenCV as a calculator. If you are more comfortable in Matlab, Python, or whatever go ahead and use those packages, porting the code I have provided as necessary.

I have written this assignment so that if you follow the directions, you will be able to complete it. However, you should try to understand why I have asked you to do certain things, and if you notice that two different operations have the same output, you should expect that I did that on purpose and try to understand why the answer is the same.

You may submit typeset solutions, or possibly the labeled text output from your program that computes everything with the matrices.

1.2.1 The Camera Matrices

I have designed a camera placement for you where the two cameras are arranged in an isosceles triangle, looking at a common fixation point. The cameras are 5 meters apart and the fixation point is 10 meters away. Camera 0 is in the canonical position at the origin with the Z axis as its optical axis. Camera 1 is translated to the right and slightly forward and rotated by approximately 29 degrees. This configuration reconciles the pictures that we have where the two cameras are turned, looking at a common fixation point, and the math, where the first camera is in the canonical position.



The fixation point is $x_0 = [0, 0, 10]$. This is the intersection of the optical axes of the cameras. Here are the internal camera parameters.

focal length	25 mm = 0.025 m
pixel size	20 μ m = 2e-5 m
$\alpha = f/p$	1250
image resolution	[1000, 1000]
principal point	[500, 500]

Here are the external parameters for camera 0 : (I is the identity matrix)

position, t_0	[0, 0, 0]
orientation	I(3)

Here are the external parameters for camera 1. They are provided with more numerical precision in the files `t1.txt` and `R1.txt`.

position, t_1	[4.8412, 0, 1.25]		
rotation	0.875	0	0.484122
	0	1	0
	-0.484122	0	0.875

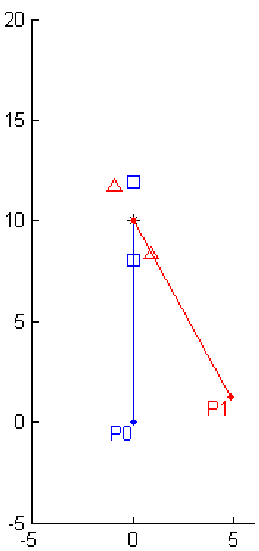
1. Write down the internal calibration matrix, K, using the internal parameters given above.
2. Write down the camera matrix, P0 for the first camera
3. Write down the camera matrix, P1, for the second camera

4. Use your camera matrices to determine the image coordinates of the fixation point. When your camera matrices are correct, the fixation point, x_0 should project to the image coordinates [500, 500] in both cameras.

1.2.2 Computing Image Coordinates

Below, I have given a collection of 3D world points arranged in a special way. I have chosen a point 3 meters above the fixation point: $[0, 3, 10]$. This point projects to the image coordinates $[500, 875]$ in both images. To help you visualize the points, I have provided a top-down drawing. Compute the image coordinates of the following points for both cameras. In the remainder of the assignment, the image coordinates will be referred to as $u_{n,0}$ for the first camera and $u_{n,1}$ for the second camera.

	x	y	z
x_1	0	3	10
x_2	0	2.4253	8.0843
x_3	0	3.5747	11.9157
x_4	0.9274	2.4253	8.3238
x_5	-0.9274	3.5747	11.6762



1.2.3 Epipolar Lines

The observant reader may notice that the points x_1, x_2, x_3 and t_0 are all co-linear, as are the points x_1, x_4, x_5, and t_1. Remember that to compute the coefficients of a 2D line connecting two points, you can represent the points in homogenous coordinates and take the cross product, leading to the coefficients [A B C] for the formula Ax + By + C = 0. I recommend normalizing the result coefficients so that B = 1 so that you can compare the equations. I have implemented a function to compute the Fundamental matrix, given two camera matrcies.

1. Compute the image coordinates of the epipole e_1, the image of the camera position of t_0 in the second camera (It will not be inside the image)
2. Compute the image coordinates of the epipole e_0, the image of the camera position t_1 in the first camera. (This will also not be in the image.) The two epipoles will not appear to be symmetric, but you should take a second to think about the business with the principal point to see why.
3. Compute the coefficients of the line through the image points $u_{2,1}$ and $u_{3,1}$
4. Compute the coefficients of the line from the epipole e_1 and the image points $u_{1,1}$
5. The line connecting the epipole e_1 and the image points $u_{1,1}$ is the epipolar line corresponding to which image points? Hint: The image points are from the first camera, and there are three.
6. Write down the Fundamental matrix
7. Use the Fundamental matrix to compute the epipolar line corresponding to $u_{1,0}$.
8. Describe the difference between inputs used to calculate the epipolar line in questions 1.3.5 and 1.3.7

1.2.4 Reconstruction

I have implemented the equation from chapter 12.2 of Hartley and Zisserman. We did the derivation in class of how to set up the matrix to use two corresponding image points together with the camera matrices in order to reconstruct the 3D point. Using the image coordinates you have computed, use the function I have written to convince yourselves that it is possible to correctly reconstruct the 3D points if you are given the corresponding image points and the camera matrices.

This is the OpenCV camera calibration / camera geometry documentation. Some of it is better documented than other parts of it. `http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html`

It is possible to recover the camera matrices, given only the 3D and 2D points. This is implemented in OpenCV, but the documentation is poor. You can implement it for yourself if you would like to see it work. There is a handout on Piazza from Chapter 7 of Hartley and Zisserman.

Finally, the Fundamental matrix can be computed from image correspondences only. This is implemented in OpenCV, but you need at least 8 points. The points provided in this assignment are all co-planar and will give a degenerate solution. If you would like to see the Fundamental Matrix computation working, you should make up some extra 3D points, compute their image coordinates, and use all the image points together.